LEVEL II

ESD-TR-81-113

MTR-8134

AD A096705

MAN-MACHINE INTERFACE (MMI) REQUIREMENTS DEFINITION AND DESIGN GUIDELINES

A PROGRESS REPORT

BY SIDNEY L. SMITH

FEBRUARY 1981

Prepared for

DEPUTY FOR TECHNICAL OPERATIONS
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
Hanscom Air Force Base, Massachusetts

Project No. 572R

Prepared by

THE MITRE CORPORATION
Bedford, Massachusetts

Contract No. F19628-80-C-0001

REVIEW AND APPROVAL

This technical report has been reviewed and is approved for publication.

MICHAEL L. WEIDNER, Capt, USAF
MMI Project Manager
Computer Engineering
Applications Division

JAMES W. NEELY, Jr., LTC, USAF
Chief, Computer Engineering
Applications Division

GEORGE M. BEZEK, Colonel, USAF
Director, Computer Systems Engineering
Deputy for Technical Operations

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: ESD-TR-81-113	2. GOVT ACCESSION NO.: AD-A096 705
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): MAN-MACHINE INTERFACE (MMI) REQUIREMENTS DEFINITION AND DESIGN GUIDELINES: A PROGRESS REPORT	5. TYPE OF REPORT & PERIOD COVERED
	6. PERFORMING ORG. REPORT NUMBER: MTR-8134
7. AUTHOR(s): Sidney L. Smith	8. CONTRACT OR GRANT NUMBER(s): F19628-80-C-0001
9. PERFORMING ORGANIZATION NAME AND ADDRESS: The MITRE Corporation, P.O. Box 208, Bedford, MA 01730	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: Project No. 572R
11. CONTROLLING OFFICE NAME AND ADDRESS: Deputy for Technical Operations, Electronic Systems Division, AFSC, Hanscom AFB, MA 01731	12. REPORT DATE: FEBRUARY 1981
	13. NUMBER OF PAGES: 81
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

DESIGN GUIDELINES
MAN-MACHINE INTERFACE
MMI
REQUIREMENTS DEFINITION

20. ABSTRACT (Continue on reverse side if necessary and ~~identify by~~ block number)

A previous report, ~~ESD-TR-80-122~~, asserted the need for man-machine interface (MMI) requirements definition and guidelines in the design of computer-based information systems. The present report extends the treatment of that topic. An initial hierarchic list of functional MMI capabilities, previously proposed for use in requirements definition, is here doubled in size to over 400 items, and has been reorganized to improve its structure. Initial design guidelines proposed for data entry functions are here revised and enlarged to include 79 items. Another (over)

DD FORM 1473, 1 JAN 73 — EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

131 guidelines are proposed for sequence control functions. A continuation of guidelines development is recommended, in collaboration with other concerned organizations and agencies.

ACKNOWLEDGMENT

This report has been prepared by the MITRE Corporation under Project No. 572R. The contract is sponsored by the Electronics Systems Division, Air Force Systems Command, Hanscom Air Force Base, Massachusetts.

TABLE OF CONTENTS

SECTION 1

INTRODUCTION

Earlier this year, with publication of MITRE Report M80-10 (Smith, 1980a), it was argued that improved techniques are needed to define requirements and provide guidance for the design of the man-machine interface (MMI) in on-line computer systems, particularly with regard to the design of operational software mediating user interaction with the system. A more extended summary of that argument is presented in Section 2 of this report, which borrows much of its wording from the original publication.

In M80-10 it was proposed that MMI requirements definition might benefit from development and use of a checklist of MMI functional capabilities. A sample list of MMI capabilities was offered in the form of a requirements matrix, illustrating how several different user tasks might have different patterns of MMI requirements. That initial list of MMI capabilities has since been enlarged and revised, as discussed in Section 3 of this report. The current version of that list is attached here as Appendix A.

In M80-10 it was further proposed that MMI design guidelines might be stated in relation to required functional capabilities. A sample set of guidelines was offered for data entry functions. Those initial guidelines for data entry have since been enlarged and reformatted, as discussed in Section 4 of this report. The currently proposed guidelines for data entry are attached here as Appendix B. In addition, a new set of guidelines for design of functions relating to sequence control is proposed here in Appendix C.

These current products represent an advance over initial proposals, but it is clear that much further work remains to be done. Recommended follow-on efforts are described in Section 5 of this report.

SECTION 2

BACKGROUND

In on-line information systems the man-machine interface (MMI) includes terminal equipment -- the various display and control devices that people use to interact with their computer tools. Also important, however, are the software programs that govern the logic of computer use, the task allocation and operating procedures that give purpose and structure to a person's interaction with a computer, the operator manuals and paper files which may have to be used in conjunction with computer processing, and other conditions of the work environment that influence job performance. A summary of the various factors that influence man-machine interface design is provided in Figure 1.

If the man-machine interface is conceived in these broad terms, to encompass all factors influencing person-system interaction, then to say that the MMI is critical to successful system operation is to state the obvious. In any automated information system, whether its work stations are used for data input, calculation, planning, management or control, effective MMI design is required for effective performance. Task analysis, review of operating procedures, equipment selection, workspace configuration, and especially MMI software design -- all must be handled with care.

The critical significance of software in MMI design was emphasized a decade ago by Parsons (1970):

> ". . . what sets data processing systems apart as a special breed? The function of each switch button, the functional arrangement among the buttons, the size and distribution of elements within a display are established not in the design of the equipment but in how the computer is programmed. Of even more consequence, the 'design' in the programs establishes the contents of processed data available to the operator and the visual relationships among the data. In combination with or in place of hardware, it can also establish the sequence of actions which the operator must use and the feedback to the operator concerning those actions."

Not only is MMI software design critical to system operation, it can also represent a significant investment of effort in system development, ranging perhaps from 10 to 50 percent or more of the operational software produced during initial system acquisition, plus software maintenance to accommodate changing operational

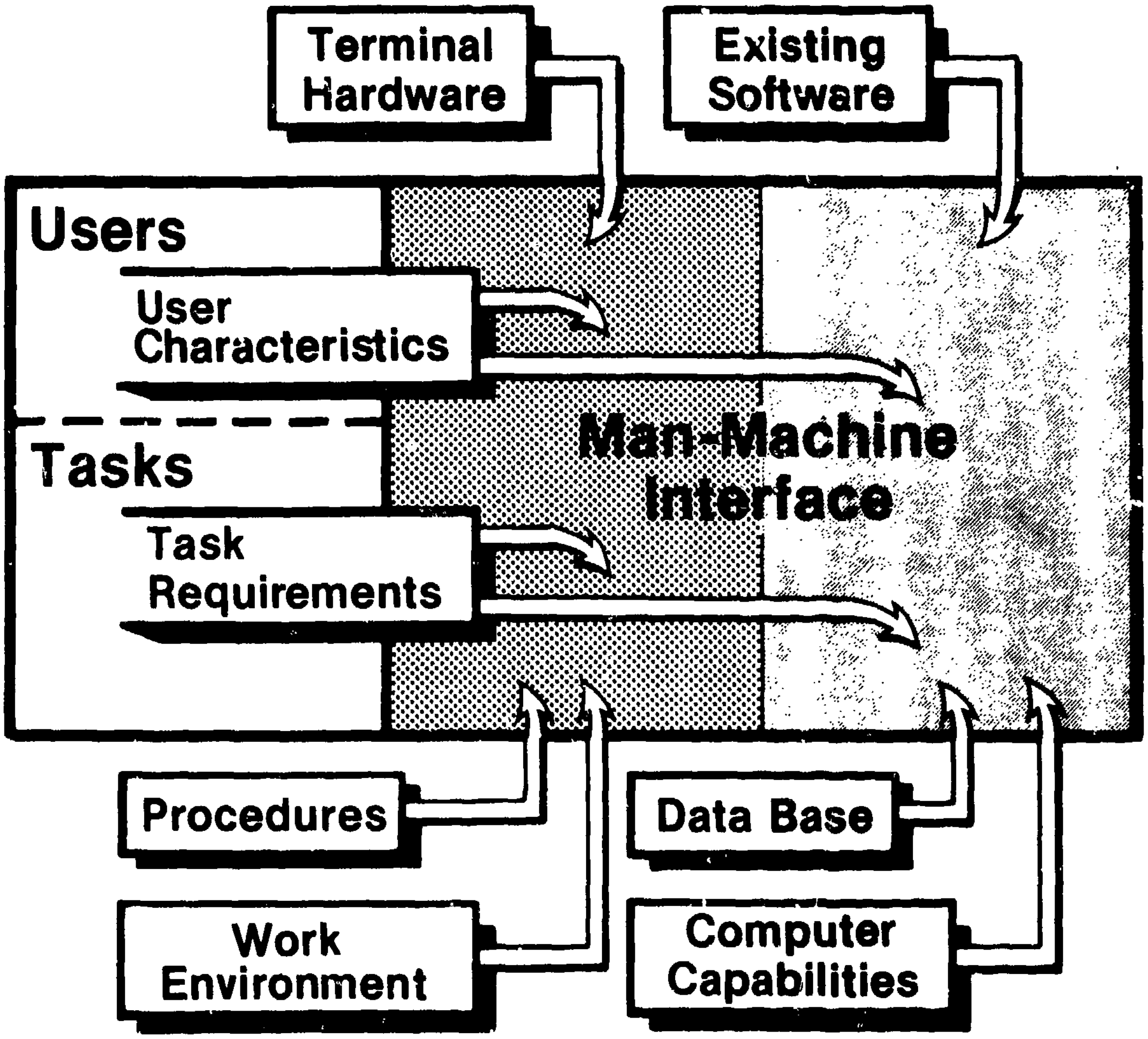

Figure 1. Factors Influencing Man-Machine Interface Design

requirements thereafter. Given the importance and the extent of MMI software, some way must be found to define MMI software requirements in system functional specification, and to provide guidelines for MMI design.

It seems fair to characterize present methods of MMI software design as art rather than science, depending more upon individual judgment than systematic application of knowledge (Ramsey and Atwood, 1979; 1980). Specifications may include only rudimentary references to MMI software design, with general statements that the system must be "easy to use". In the absence of more effective guidance, both the design and implementation of MMI software may become the responsibility of programmers unfamiliar with operational requirements. MMI software may be produced slowly. Detection and correction of design flaws may occur only after system prototyping, when software changes are difficult to make.

As an art, MMI design is best practiced by experts, by specialists experienced in the human engineering of man-computer systems. But such experts are not always available to help guide system acquisition, and certainly cannot guide every step of MMI design at first hand. What seems needed is some way to embody expert judgment in the form of explicit procedures and guidelines for MMI design.

Present human engineering standards and design guide books are of little use to the software designer, being oriented toward hardware design ("knobs and dials") and physical safety, and including only token references to software. A recent bibliography of the literature on human factors in computer systems describes 564 reports, but identifies only 17 as offering design guidelines (Ramsey, Atwood and Kirshbaum, 1978). So here is a significant problem: guidelines for MMI design are needed, but few are available.

SECTION 3

REQUIREMENTS DEFINITION

The first step in MMI software design is to decide what is needed. Once MMI requirements are determined, then design guidelines can be tailored to anticipated needs. MMI requirements definition must consider the characteristics of the people who will use the system, the information handling requirements of their jobs, and the functional capabilities of the MMI that are needed in order to perform those jobs.

USER CHARACTERISTICS

Most information systems are used by a broad mix of people of different types. For any one category of user, individual differences in skill may be considerable. In military systems, because of systematic rotation of job assignments for personnel, today's nai e user becomes next year's expert, then to be replaced by another beginner. This is true of many non-military systems as well. Such regular personnel turnover implies the need for flexible job aids in MMI design, which can provide optional help to the novice user but which can be bypassed by the expert.

For the developers and designers of information systems, it may help to postulate some general characteristics of prospective users. We can assume that users will be intelligent men and women with their own special skills. These people will not necessarily be knowledgeable about computer technology, may have little time to learn complicated interface procedures, and will have different degrees of familiarity with the system. Being human, these people will sometimes make mistakes, especially when working under pressure, and good MMI design must take that into account.

These people are usually motivated toward effective job performance in the face of operational demands. They will regard automated data processing as a tool to aid job performance, with little curiosity about the internal mechanisms of computing machines. Users will tend to judge the entire system on the basis of their personal experience with the MMI. If the MMI is efficient and easy to use, they will like the system. But users will be impatient and critical when handicapped by a clumsy interface design.

TASK ANALYSIS

Fundamental to MMI design is the analysis of user tasks. This analysis must begin with the mission requirements of a proposed system, which state the basic objectives to be accomplished. These mission requirements are then elaborated and translated, taking into account the proposed operational employment concept, environmental, technological and fiscal constraints, to define the system operational requirements.

Operational requirements imply the performance of various identifiable functions -- data sensing, data transmission, data processing, etc. Analysis of those functions, in turn, establishes more specific data processing requirements -- what data must enter the system, what data must be stored, what combinations and transformations of data are required, what kinds of information should result from that processing.

These data processing functions imply the specification of tasks to accomplish particular ends. Some tasks may be performed entirely by machine and thus affect MMI design only indirectly if at all. Because of the critical role of human judgment, however, and the fact that much of the data to be processed must be generated and/or evaluated by system users, many tasks will involve joint performance by man and machine.

Most user tasks can be partitioned further into identifiable subtasks. Those subtasks in turn are often designed as a series of simple, discrete transactions, such as entry of a single item of data. (As defined here, note that a transaction is the smallest functional "molecule" of man-machine interaction, and does not denote an extended task sequence.) Each identified task, subtask and transaction will imply the need for particular functional capabilities in MMI design.

FUNCTIONAL CAPABILITIES

One could imagine developing a long list of possible MMI capabilities for consideration in system design. A first version of such a list was published in MITRE Report M80-10 (Smith, 1980a). In that initial list there were 210 base-level items, plus another 100 supraordinate labels, comprising some nine pages. As the initial list has been applied in actual MMI requirements definition, it has grown in length to 473 base-level items on 18 pages. That current version of the list is attached as Appendix A to this report.

Presumably growth will continue as the MMI capabilities list is used in further applications, each emphasizing particular aspects of the MMI with consequent elaboration of that portion of the list. It seems probable, however, that the growth rate will decline as the list becomes more comprehensive, and that eventually the list will reach a stable form and size.

How can such a list be used in MMI requirements definition? It seems clear that any particular capability in the list may be required for some tasks but not for others. As an example, a capability for pointing (1.1 Position Designation) is essential for tasks involving frequent interaction with a graphic display. For an on-line editing task, where the user must designate arbitrary portions of displayed text for correction, pointing would clearly be useful. For a task involving sequential selections among computer-displayed options, pointing would probably be useful, although other design alternatives such as multi-function keys might do nearly as well. For many other tasks pointing may not be needed at all.

One could imagine taking a list of MMI capabilities and estimating whether each item on the list is required for the performance of a particular user task. The format illustrated in Appendix A permits checking whether each capability is estimated to be essential, potentially useful, or not needed for effective task performance. Such a checklist can produce a simple "profile" of MMI functional requirements. In the future it may be possible to make more refined MMI requirements estimates.

As different user tasks are analyzed, their separate lists of requirements estimates could be combined in a large table, here called a "matrix", with MMI functional capabilities listed as the row labels, and the requirements for different tasks recorded in columns. What advantages could a requirements matrix offer? First, it seems clear that such a matrix, even in its early, rudimentary stages, should help to provide general perspective and serve as a framework to structure judgment in determining MMI requirements. As noted above, the row labels in the matrix would constitute a checklist to remind system developers of MMI capabilities that should be considered.

With further development, as the MMI requirements matrix expands to include more functional capabilities and a manifold range of tasks, it can embody the experience accumulated in a variety of system acquisition programs. The MMI requirements matrix might eventually include entries for perhaps a thousand or more specific functional capabilities and perhaps as many as 50 characteristic user tasks. It should then be possible to draw on this cumulative judgment instead of having to re-invent an MMI from scratch for each new system.

By its nature, the MMI requirements matrix cannot be created all at once, but will grow gradually through accretion, as new capabilities and tasks are added and old ones are further analyzed and subdivided. After several years of experience using the matrix, some estimate might be made of the eventual success of this approach to MMI requirements definition. To gain that experience, a beginning must be made in current system acquisition programs. Such an effort is now being undertaken.

SECTION 4

DESIGN GUIDELINES

Once MMI requirements have been defined in terms of the functional capabilities needed to perform identified user tasks, then it should prove possible to specify a set of design guidelines "tailored" to those requirements. Appropriate guidelines might be adopted from various published sources, or might be developed on an ad hoc basis where no published guidance seems suitable. Guidelines are presented here for MMI functions relating to data entry and sequence control.

DATA ENTRY

As an initial sample of what might be developed, a set of design guidelines for MMI capabilities relating to data entry functions was published in MITRE Report M80-10 (Smith, 1980a). That initial set listed some 65 guidelines on 10 pages, which were adapted from the best available guidelines, those published by Engel and Granda (1975) and by Pew and Rollins (1975).

During the past several months, the initial set of design guidelines of data entry has been augmented in size to 79 items. That current set is attached as Appendix B to this report.

In comparison with its earlier version, the list of guidelines has been modified in format to correspond with concurrent changes in the structure of the MMI capabilities list.

The guidelines format has also been modified to permit separate listing of illustrative examples, exceptions, comments and references. Separate listing of such supplemental items serves to clarify their status, and will permit their optional inclusion in any future guidelines publication.

In this revised format, certain significant deficiencies are evident in the current set of guidelines. For example, there are no guidelines listed here for entry of textual data, although surely some could be developed. There are also no guidelines here for graphic data entry. That deficiency may be remedied next year as a result of research soon to be reported by Granda (1980). With continued effort, a comprehensive list of data entry guidelines should be attainable.

SEQUENCE CONTROL

As a next step in enlarging the domain of MMI design application, a set of 131 guidelines pertaining to functional capabilities for sequence control is attached as Appendix C to this report. Here sequence control refers to the means by which the user of an on-line information system initiates and concludes individual transactions.*

The guidelines for sequence control proposed in Appendix C represent an initial set which will undoubtedly benefit from future additions and modifications. These guidelines have again been adapted from the best available published reports, those by Engel and Granda (1975) and by Pew and Rollins (1975), but incorporate further material based on independent judgment.

Comments and criticism of these proposed guidelines will be welcome. Where guidelines must be based on judgment rather than experimental data, which is now often the case, a broad range of judgment is needed to ensure that recommended design guidance is sound.

---

* A more extended discussion of sequence control has been presented in a previous report (Smith, 1980a).

SECTION 5

FOLLOW-ON EFFORT

These propose tools for MMI requirements definition and design guidance will need continued development, and application in system acquisition programs, to determine their practical value. Coordinated exchange of information with other groups involved in related efforts will be needed to support the long-range establishment of MMI design standards.

CONTINUED DEVELOPMENT

As noted in preceding sections of this report, further work will be necessary to develop a more complete list of functional capabilities for MMI requirements definition, and to develop a more complete set of MMI design guidelines. It will also be necessary to develop more effective means of documenting MMI design.

Requirements

The present list of MMI functional capabilities (Appendix A) represents a significant improvement over its initial version, but is still by no means complete. Two approaches appear feasible for further improvement of the capabilities list: continued application in system acquisition, and solicitation of expert judgment. These two approaches are discussed further below.

Guidelines

For the present, guidelines are not available for all MMI functional areas. A continuing effort will be needed to develop a more complete set of MMI guidelines, in coordination with the related work of other groups. Those guidelines we do have are more often based on judgment than quantitative performance measures, and broader judgment should be brought to bear on the review of proposed guidelines. Meanwhile it is clear that whatever MMI guidelines are developed at this stage can only be regarded as tentative, to be used on an interim basis until experience is gained in their trial application.

Documentation

Improved MMI requirements definition and the development of guidelines, although essential, will not be sufficient to ensure effective interface design. Some means must be found to document requirements and guidelines in a form that will facilitate design review and communication among the developers and eventual users of any information system. It is the MMI design that most clearly characterizes a system to its users. Careful documentation can help assure that the user's view of the system will be congruent with that of the designer. This topic has been discussed by Pew, Vittal and Sidner (1980), who suggest that a specialized language may need to be developed for documenting MMI design. Whether or not this is true, it is clear that there is need for exploration of better means of MMI design documentation.

APPLICATION

Looking ahead, our eventual goal must be to express the knowledge of the behavioral scientist, the human engineering specialist, and the user of on-line information systems, in a form useful to designers. Proposed guidelines must be evaluated in practical application as a collaborative effort among all concerned with improving system design. Design guidelines that appear useful in practice would be retained, others modified, and new guidelines added as necessary to meet identified requirements.

The current Army effort to develop and apply MMI design guidelines for battlefield information systems has been reported by Sidorsky and Parrish (1980). Trial application in Air Force command systems has been recommended by Smith (1980a), and a beginning has been made. Application of MMI design guidelines has also been advocated by NASA (1979). More efforts of this kind will be needed to broaden the range of application.

INFORMATION EXCHANGE

At several points in the preceding discussion there has been cited the need for coordination with other agencies to achieve an effective follow-on effort. Coordination must be based on information exchange. A beginning has been made by promoting wide distribution of MITRE Report M80-10, and by publicizing the concepts discussed in that report (Smith, 1980b; 1980c; 1980d). Such efforts will continue. Beyond that, it will be necessary to follow current work by other groups concerned with MMI requirements definition and guidelines development in order to encourage more general application of this approach.

The most comprehensive on-going program that has been publicly reported is the Army-sponsored guidelines effort mentioned above (Sidorsky and Parrish, 1980). That program should produce valuable results within another two or three years. Meanwhile, both Navy and Air Force sponsors are planning to encourage programmatic research directed toward problems of MMI design. The Office of Naval Research has publicized special research opportunities in this area. The Air Force Human Resources Laboratory and the Aerospace Medical Research Laboratory both plan to institute related research programs in their respective areas of interest.

Various industrial organizations, including IBM, ATT, Xerox and others, are currently undertaking efforts to improve MMI design. Some of this industrial effort is directed toward improving product design and user acceptance. Some is directed toward internal standardization of MMI design to facilitate more effective in-house use of information systems. In either case there is concern for protecting proprietary interests, and reluctance at this stage to publicize in-house guidelines as recommendations for more general MMI design standards.

Current academic research tends to concentrate on applications of advanced technology, such as speech input to computers using natural language. In academic research both the experimenters and their user subjects are often computer specialists or other people keenly interested in computer systems. To the extent that this is true, the more mundane concerns of task-oriented users in commonplace work settings are overlooked or at least discounted. Practical MMI design guidelines are unlikely to result from such studies, although appropriately directed academic research could be of value in measuring the quantitative performance gains achieved by proposed improvements to MMI design.

Some means should be found to bring together people concerned with MMI requirements definition and guidelines development. It is recommended that an effort be made to establish a working group, or "experts panel", in which representatives of user agencies, industrial designers and academic researchers exchange information on a regular basis in order to speed the development and critical review of MMI design guidelines.

DESIGN STANDARDS

The practical evaluation of proposed design guidelines may have to extend over a period of many years, since it is in some degree linked to the pace of system acquisition. Such an extended effort may well be justified, however, in view of the potential long-term benefits. There is one important factor to consider in this regard, which is the relative stability of human engineering guidelines. Standards for hardware design may change as each new generation of equipment becomes available. But people change hardly at all, from one generation to the next, in terms of their basic information processing abilities and limitations.

This relative invariance of people with respect to technology offers a significant advantage: if MMI guidelines can be expressed in terms of human characteristics rather than transient technology, a design standard of enduring value will be established. It may be true that such guidelines can be established only slowly. If so, it is important that current efforts be continued as required to complete the job.

One individual, of course, cannot set design standards. That must be a collaborative effort among many contributors, as described above. The time is at hand when such an effort may prove productive. As one indication of this, the current draft of a planned revision to MIL-STD-1472B (1974) contains for the first time some guidance (nine pages) pertaining to the design of MMI software for on-line information systems. This is clearly inadequate in comparison to the need, but it is a start. With dedicated effort, it should be possible to develop a comprehensive set of guidelines for MMI design within the next several years, for review, comment, modification as needed, and eventual adoption as an agreed design standard several years thereafter.

Achieving agreed MMI design standards is only a necessary first step toward improved system design. The integrated approach to requirements definition, functional specification and design guidance, recommended here, offers the potential means for increasing productivity in MMI software design. In the long run, the most significant value of standardizing MMI functions may be to permit the use of modular software components as building blocks for system development. With consequent benefits for system operation as well as system development, that is a goal worth working toward.

REFERENCES

Engel, S. E. and Granda, R. E. Guidelines for man/display interfaces, Technical Report TR 00.2720. Poughkeepsie, New York: IBM, December 1975.

Granda, R. E. Man/machine design guidelines for the use of screen display terminals. Paper presented at the 24th Annual Meeting of the Human Factors Society, Los Angeles, California, 13-16 October 1980.

MIL-STD-1472B. Military standard: human engineering design criteria for military systems, equipment and facilities. Washington: Department of Defense, 31 December 1974.

National Aeronautics and Space Administration (NASA). Spacelab experiment computer application software (ECAS) display design and command usage guidelines, Report MSFC-PROC-711. George C. Marshall Space Flight Center, Alabama, January 1979.

Parsons, H. M. The scope of human factors in computer-based data processing systems. Human Factors, 1970, 12(2), 165-175.

Pew, R. W. and Rollins, A. M. Dialog specification procedures, Report 3129 (revised). Cambridge, Massachusetts: Bolt Beranek and Newman, 1975.

Pew, R. W., Vittal, J. J. and Sidner, C. Man-machine interface design documentation: Representing the user's model of a system. Paper presented at the 24th Annual Meeting of the Human Factors Society, Los Angeles, California, 13-16 October 1980.

Ramsey, H. R. and Atwood, M. E. Human factors in computer systems: a review of the literature, Technical Report SAI-79-111-DEN. Englewood, Colorado: Science Applications, Inc., September 1979. (NTIS No. ADA075679)

Ramsey, H. R. and Atwood, M. E. Man-machine interface design guidance: state of the art. Paper presented at the 24th Annual Meeting of the Human Factors Society, Los Angeles, California, 13-16 October 1980.

Ramsey, H. R., Atwood, M. E. and Kirshbaum, P. J. A critically annotated bibliography of the literature of human factors in computer systems, Technical Report SAI-78-070-DEN. Englewood, Colorado: Science Applications, Inc., May 1978. (NITS No. ADA058081)

Sidorsky, R. C. and Parrish, R. N. Guidelines and criteria for man-machine interface design of battlefield automated systems. Paper presented at the 24th Annual Meeting of the Human Factors Society, Los Angeles, California, 13-16 October 1980.

Smith, S. L. Requirements definition and design guidelines for the man-machine interface in C3 system acquisition, Report M80-10. Bedford, Massachusetts: The MITRE Corporation, 15 April 1980. (a) (Note: This report was published by the Air Force Electronic Systems Division as Technical Report ESD-TR-80-122, dated June 1980, and may be obtained from NTIS as Document No. ADA087258.)

Smith, S. L. Requirements definition and design guidelines for the man-machine interface. Paper presented at National Aerospace and Electronics Conference, Dayton, Ohio, 20-22 May 1980. (b)

Smith, S. L. Man-machine interface requirements definition: task demands and functional capabilities. Paper presented at Human-Machine Systems Symposium, International Conference on Cybernetics and Society, Boston, Massachusetts, 8-10 October 1980. (c)

Smith, S. L. Man-machine interface requirements definition: task demands and functional capabilities. Paper presented at the 24th Annual Meeting of the Human Factors Society, Los Angeles, California, 13-16 October 1980. (d)

APPENDIX A: MMI REQUIREMENTS CHECKLIST

Task ______________________ Reviewer ______ Date ______

MMI Capability	Requirement Estimate* E	U	N	Comment
1.0 DATA ENTRY/INPUT				
1.1. Position Designation				
1. arbitrary positions	.	.	.	-------------
1 discrete	__	__	__	
2 continuous	__	__	__	
2. predefined positions	.	.	.	-------------
1. HOME	.	.	.	
1 upper left	__	__	__	
2 center	__	__	__	
3 lower right	__	__	__	
4 other	__	__	__	
2 command entry area	__	__	__	
3 end of file	__	__	__	
4 other	__	__	__	
3. incremental positions	.	.	.	-------------
1. by character	.	.	.	
1 right	__	__	__	
2 left	__	__	__	
3 up	__	__	__	
4 down	__	__	__	
2. by interval (TAB)	.	.	.	
1 horizontal	__	__	__	
2 vertical	__	__	__	
3. by other features	.	.	.	
1 word	__	__	__	
2 line	__	__	__	
3 paragraph	__	__	__	
4 other	__	__	__	
1.2. Direction Designation				
1 vector rotation	__	__	__	-------------
2 sequential pointing	__	__	__	-------------
3 numeric entry	__	__	__	-------------
4 other	__	__	__	-------------

* E = Essential, U = Useful, N = Not Needed

MMI Capability	E	U	N	Comment
1.3. Text				
1. predefined format	.	.	.	- - - - - - - - - - - - - - -
1. select	.	.	.	
1 header	___	___	___	
2 paragraph	___	___	___	
3 page	___	___	___	
4 other	___	___	___	
2. enter		.	.	
1 insert	___	___	___	
2 append	___	___	___	
3 change	___	___	___	
4. delete	.	.	.	
1 character	___	___	___	
2 line	___	___	___	
3 paragraph	___	___	___	
4 page	___	___	___	
5 other	___	___	___	
2. user-defined format	.	.	.	- - - - - - - - - - - - - - -
1 create	___	___	___	
2 enter	___	___	___	
3 change	___	___	___	
4. delete	.	.	.	
1 character	___	___	___	
2 line	___	___	___	
3 paragraph	___	___	___	
4 page	___	___	___	
5 other	___	___	___	
1.4. Data Forms				
1. predefined format	.	.	.	- - - - - - - - - - - - - - -
1. select	.	.	.	
1 header	___	___	___	
2 field	___	___	___	
3 section	___	___	___	
4 page	___	___	___	
5 entire form	___	___	___	
2 enter	___	___	___	
3 change	___	___	___	
4. delete	.	.	.	
1 character	___	___	___	
2 field (BACKUP)	___	___	___	
3 section (CANCEL)	___	___	___	
4 page (RESTART)	___	___	___	
5 other	___	___	___	

MMI Capability	E	U	N	Comment
1.4.2. user-defined format	.	.	.	---------------
1 create	___	___	___	
2 enter	___	___	___	
3 change	___	___	___	
4. delete	.	.	.	
1 character	___	___	___	
2 field (BACKUP)	___	___	___	
3 section (CANCEL)	___	___	___	
4 page (RESTART)	___	___	___	
5 other	___	___	___	
1.5. Tabular Data				
1. predefined format	.	.	.	---------------
1. select	.	.	.	
1 header	___	___	___	
2 object (row)	___	___	___	
3 property (column)	___	___	___	
4 page	___	___	___	
5 other	___	___	___	
2 enter	___	___	___	
3 change	___	___	___	
4. delete	.	.	.	
1 character	___	___	___	
2 row	___	___	___	
3 column	___	___	___	
4 page	___	___	___	
5 other	___	___	___	
2. user-defined format	.	.	.	---------------
1 create	___	___	___	
2 enter	___	___	___	
3 change	___	___	___	
4. delete	.	.	.	
1 character	___	___	___	
2 row	___	___	___	
3 column	___	___	___	
4 page	___	___	___	
5 other	___	___	___	

MMI Capability	E	U	N	Comment
1.6. Graphic Data				
1. predefined format	.	.	.	- - - - - - - - - - - - - -
1. select	.	.	.	
1. plot type	.	.	.	
1 geographic	___	___	___	
2 line graph	___	___	___	
3 bar graph	___	___	___	
4 pie chart	___	___	___	
5 other	___	___	___	
2 background/map	___	___	___	
3 data category	___	___	___	
4 symbol	___	___	___	
5 other	___	___	___	
2 enter	___	___	___	
3 change	___	___	___	
4. delete	.	.	.	
1 point	___	___	___	
2 symbol	___	___	___	
3 drawn line	___	___	___	
4 data category	___	___	___	
5 other	___	___	___	
2. user-defined format	.	.	.	- - - - - - - - - - - - - -
1 create	___	___	___	
2 enter	___	___	___	
3 change	___	___	___	
4. delete	.	.	.	
1 point	___	___	___	
2 symbol	___	___	___	
3 drawn line	___	___	___	
4 data category	___	___	___	
5 other	___	___	___	
1.7. Data Validation				
1. required entry	.	.	.	- - - - - - - - - - - - - -
1 immediate	___	___	___	
2 deferrable	___	___	___	
2. length of entry	.	.	.	- - - - - - - - - - - - - -
1 fixed	___	___	___	
2 maximum	___	___	___	
3 minimum	___	___	___	
3. content of entry	.	.	.	- - - - - - - - - - - - - -
1 numeric	___	___	___	
2 alphabetic	___	___	___	
3 alphanumeric	___	___	___	
4 defined codes	___	___	___	
5 other	___	___	___	

MMI Capability	E	U	N	Comment
1.7.4. comparative checks	.	.	.	_______________
1 equal to	___	___	___	
2 greater than	___	___	___	
3 less than	___	___	___	
4 IF. .THEN	___	___	___	
5 other	___	___	___	
5. default entry	.	.	.	_______________
1 predefined	___	___	___	
2 user-defined	___	___	___	
1.8. Other Data Processing				
1. file management	.	.	.	_______________
1 merging/linking	___	___	___	
2. cross-file update	.	.	.	
1 automatic	___	___	___	
2 by request	___	___	___	
2. derived statistics	.	.	.	_______________
1 total	___	___	___	
2 mean	___	___	___	
3 median	___	___	___	
4 range (of values)	___	___	___	
5 other	___	___	___	
3. other computation	.	.	.	_______________
1 date/time	___	___	___	
2 grid conversion	___	___	___	
3 azimuth	___	___	___	
4 range (distance)	___	___	___	
5 other	___	___	___	
1.9. Design Change				
1. type	.	.	.	_______________
1 file formats	___	___	___	
2 entry formats	___	___	___	
3 item specification	___	___	___	
4 data processing	___	___	___	
5 other	___	___	___	
2. implementation	.	.	.	_______________
1. on-line	.	.	.	
1 by transaction	___	___	___	
2 by software change	___	___	___	
2 off-line	___	___	___	

MMI Capability	E	U	N	Comment
2.0 DATA DISPLAY/OUTPUT				
2.1. Data Type				
1. text	.	.	.	
1 formatted				
2 unformatted				
2 data forms				
3 tabular				
4 graphic				
5 combination				
2.2. Data Density				
1. text	.	.	.	
1 high (> 1000 char.)				
2 moderate				
3 low (< 600 char.)				
2. data forms	.	.	.	
1 high (> 600 char.)				
2 moderate				
3 low (< 300 char.)				
3. tabular	.	.	.	
1 high (> 600 char.)				
2 moderate				
3 low (< 300 char.)				
4. graphic	.	.	.	
1 high (> 300 char.)				
2 moderate				
3 low (< 100 char.)				
2.3. Data Aggregation				
1 summary display				
2 grouped items				
3 individual items				
2.4. Data Coding				
1. variables/dimensions	.	.	.	
1 many				
2 moderate				
3 few				
2. categories	.	.	.	
1 many (> 20)				(alphanumeric)
2 moderate (8-20)				(symbol)
3 few (3-7)				(color)
4 just two				(size, brightness)
3. criticality	.	.	.	
1 high				(redundant coding)
2 moderate				(separate coding)
3 low				

MMI Capability	E	U	N	Comment
2.4.4. code format	.	.	.	
1 predefined				
2 user-defined				
2.5. Display Generation				
1. automatic	.	.	.	
1 predefined				
2 user-defined				
2 by request				
3. printout	.	.	.	
1 from display				
2 from file				
3 selected data				
2.6. Data Selection				
1. file	.	.	.	
1 name				
2 number				
3 other				
2. file subset	.	.	.	
1 name				
2 number				
3 page				
4 other				
3. data item	.	.	.	
1 name				
2 number				
3 other				
4. data category	.	.	.	
1 time period				
2 area				
3 other				
5 combinatorial logic				
2.7. Display Suppression				
1. automatic	.	.	.	
1 timeout/fading				
2 other				
2. by request	.	.	.	
1 data item				
2. data category	.	.	.	
1 time period				
2 area				
3 other				
3 all data (CLEAR)				
3. duration	.	.	.	
1 continuing				
2 temporary				

MMI Capability	E	U	N	Comment
2.8. Display Coverage				
1. displacement	.	.	.	
1. page (text, tabular)	.	.	.	
1 forward	___	___	___	
2 back	___	___	___	
2. scroll (text)	.	.	.	
1 up	___	___	___	
2 down	___	___	___	
3 offset (graphic)	___	___	___	
4 return/recenter	___	___	___	
2. expansion	.	.	.	
1. discrete increments	.	.	.	
1 predefined	___	___	___	
2 user-defined	___	___	___	
2. continuous	.	.	.	
1 zoom in	___	___	___	
2 zoom out	___	___	___	
3 return/normalize	___	___	___	
3. partitioning	.	.	.	
1 fixed windows	___	___	___	
2. variable windows	.	.	.	
1 automatic	___	___	___	
2 by request	___	___	___	
3 multiple displays	___	___	___	
2.9. Display Update				
1 automatic	___	___	___	
2 by request	___	___	___	
3. rate	.	.	.	
1 normal	___	___	___	(event driven)
2 fast	___	___	___	(time compression)
3 slow	___	___	___	
4 freeze	___	___	___	("stop action")
2.10. Design Change				
1. type	.	.	.	
1 file format	___	___	___	
2 display format	___	___	___	
3 other	___	___	___	
2. implementation	.	.	.	
1. on-line	.	.	.	
1 by transaction	___	___	___	
2 by software change	___	___	___	
2 off-line	___	___	___	

MMI Capability	E	U	N	Comment
3.0 SEQUENCE CONTROL				
3.1. Dialogue Type				
1 question and answer				
2 form filling				
3 menu selection				
4 function keys				
5 command language				
6 query language				
7 natural language				
8 graphic interaction				
3.2. Transaction Selection				
1 general OPTIONS				
2 implicit options				
3. contingent	.	.	.	(step-specific options)
1 automatic				
2 by request				
4 stacked commands				
5 linked commands				(macros)
3.3. Interrupt				
1 CANCEL				
2 BACKUP				
3 RESTART				
4 ABORT				
5 END				
3.4. Context Definition				
1 automatic				
2. by request	.	.	.	
1 user command				
2 data category				
3 data item				
3.5. Error Management				
1 command validation				
2 explicit ENTER				
3 CONFIRM protection				
4 direct error correction				

MMI Capability	E	U	N	Comment
3.6. Alarms				
1. alarm definition	.	.	.	(task-related) _ _ _ _ _ _ _
1 predefined	___	___	___	
2. user-defined	.	.	.	
1 variables	___	___	___	
2 categories	___	___	___	
2. alarm acknowledgment	.	.	.	_ _ _ _ _ _ _ _ _ _ _ _ _ _
1. automatic	.	.	.	
1 routine	___	___	___	(timeout)
2 implicit action	___	___	___	(seen)
3 other	___	___	___	
2. user action	.	.	.	
1 predefined	___	___	___	
2 user-defined	___	___	___	
3 override	___	___	___	
3.7. Design Change				
1. type	.	.	.	_ _ _ _ _ _ _ _ _ _ _ _ _ _
1 available options	___	___	___	
2 sequence logic	___	___	___	
3 data processing	___	___	___	
4 other	___	___	___	
2. implementation	.	.	.	_ _ _ _ _ _ _ _ _ _ _ _ _ _
1. on-line	.	.	.	
1 by transaction	___	___	___	
2 by software change	___	___	___	
2 off-line	___	___	___	

MMI Capability	E	U	N	Comment
4.0 USER GUIDANCE				
4.1. Status Information				
1. operability	.	.	.	
1 local work station				
2. system	.	.	.	
1 equipment				
2 data files				
3 functions				
3 external				
2 current users				
3 current load				(response time)
4 other notices				
5. time signals	.	.	.	
1 continuous				
2 periodic				
3 by request				
4 date/time				
6. alarm signals	.	.	.	(system-related)
1 variables/dimensions				
2 categories				
4.2. Routine Feedback				
1. input	.	.	.	
1 data entry				
2 data change				
3 data deletion				
2. output	.	.	.	
1 data displayed				
2 partial display				(exceeds capacity)
3 data not available				
3. sequence control	.	.	.	
1 requested transaction				
2 changed context				
3 other				
4.3. Error Feedback				
1 error type				
2 correction procedure				
3 alert signals				
4 cursor position				

MMI Capability	E	U	N	Comment
4.4. Job Aids				
1. automatic prompts	.	.	.	---------------
1 fixed messages	___	___	___	
2. contingent on input	.	.	.	
1 command selection	___	___	___	
2 context change	___	___	___	
3 data entry	___	___	___	
3. command aiding	.	.	.	
1 branching options	___	___	___	
2 disambiguation	___	___	___	
3 other	___	___	___	
4. cursor position	.	.	.	
1 command entry area	___	___	___	
2 data entry field	___	___	___	
3 error location	___	___	___	
4 off screen	___	___	___	
5 other	___	___	___	
2. by request	.	.	.	---------------
1 command index	___	___	___	
2 data index	___	___	___	
3 HELP, EXPLAIN	___	___	___	
4 on-job training	___	___	___	
5 other	___	___	___	(simulation, etc.)
3. instructional level	.	.	.	---------------
1 novice users	___	___	___	
2. transitional users	.	.	.	
1 by time of use	___	___	___	
2 by measured skill	___	___	___	
3 other	___	___	___	
3 expert users	___	___	___	
4 mixed user skills	___	___	___	
4.5. User Records				
1 transactions	___	___	___	---------------
2 files accessed	___	___	___	---------------
3 programs used	___	___	___	---------------
4 errors made	.	.	.	---------------
1 data entry/change	___	___	___	
2 sequence control	___	___	___	
5 help requested	___	___	___	---------------
6 other	___	___	___	---------------

MMI Capability	E	U	N	Comment
4.6. Design Change				
1. type	.	.	.	- - - - - - - - - - - - - -
1 status information	___	___	___	
2 alarms/alerts	___	___	___	
3 error messages	___	___	___	
4 prompts	___	___	___	
5 auxiliary help	___	___	___	
6 training aids	___	___	___	
7 other	___	___	___	
2. implementation	.	.	.	- - - - - - - - - - - - - -
1. on-line	.	.	.	
1 by transaction	___	___	___	
2 by software change	___	___	___	
2 off-line	___	___	___	

MMI Capability	E	U	N	Comment
5.0 DATA TRANSFER/COMMUNICATION				
5.1. Data Transmission				
1. source	.	.	.	- - - - - - - - - - - - - -
1 from display	___	___	___	
2 from own files	___	___	___	
2. destination	.	.	.	- - - - - - - - - - - - - -
1. to files	.	.	.	
1 own	___	___	___	
2 other users	___	___	___	
2 to other terminals	___	___	___	
3 to printer	___	___	___	
4 to other device	___	___	___	
5 external	___	___	___	
3. data type	.	.	.	- - - - - - - - - - - - - -
1. text	.	.	.	
1. formatted	.	.	.	
1 messages	___	___	___	
2 documents	___	___	___	
2 unformatted	___	___	___	
2 data forms	___	___	___	
3 tabular	___	___	___	
4 graphic	___	___	___	
5 alarm/alert signals	___	___	___	
4. transmission control	.	.	.	- - - - - - - - - - - - - -
1. data specification	.	.	.	
1. by display name	.	.	.	
1 all	___	___	___	
2 designated part	___	___	___	
2. by file name	.	.	.	
1 all	___	___	___	
2 designated part	___	___	___	
3. by data name	.	.	.	
1 category	___	___	___	
2 item	___	___	___	
2. initiation	.	.	.	
1. automatic	.	.	.	
1 continuous	___	___	___	
2 periodic	___	___	___	
3 contingent	___	___	___	(on transaction)
2 by request	___	___	___	

MMI Capability	E	U	N	Comment
5.1.5. feedback	.	.	.	_______________
1. category	.	.	.	
1 initiated	___	___	___	
2 confirmed	___	___	___	
3 failed	___	___	___	
2. specification	.	.	.	
1. automatic	.	.	.	
1 predefined	___	___	___	
2 user-defined	___	___	___	
2 by request	___	___	___	
6. queueing	.	.	.	_______________
1 automatic	___	___	___	
2 by request	___	___	___	
7. record keeping	.	.	.	_______________
1. log	.	.	.	
1 automatic	___	___	___	
2 by request	___	___	___	
2. journal	.	.	.	
1 automatic	___	___	___	
2 by request	___	___	___	
5.2. Data Receipt				
1. source	.	.	.	_______________
1 from other files	___	___	___	
2 from other terminals	___	___	___	
3 external	___	___	___	
2. destination	.	.	.	_______________
1 to display	___	___	___	
2 to own files	___	___	___	
3 to printer	___	___	___	
4 to other device	___	___	___	
3. data type	.	.	.	_______________
1. text	.	.	.	
1. formatted	.	.	.	
1 messages	___	___	___	
2 documents	___	___	___	
2 unformatted	___	___	___	
2 data forms	___	___	___	
3 tabular	___	___	___	
4 graphic	___	___	___	
5 alarm/alert signals	___	___	___	

MMI Capability	E	U	N	Comment
5.2.4. transmission control	.	.	.	- - - - - - - - - - - - - -
1. data specification	.	.	.	
1 by source	_	_	_	
2. by file name	.	.	.	
1 all	_	_	_	
2 designated part	_	_	_	
3. by data name	.	.	.	
1 category	_	_	_	
2 item	_	_	_	
2. initiation	.	.	.	
1. automatic	.	.	.	
1 continuous	_	_	_	
2 periodic	_	_	_	
3 contingent	_	_	_	(on transaction)
2 by request	_	_	_	
5. notification	.	.	.	- - - - - - - - - - - - - -
1. category	.	.	.	
1 source	_	_	_	
2 type	_	_	_	
3 priority	_	_	_	
2. specification	.	.	.	
1. automatic	.	.	.	
1 predefined	_	_	_	
2 user-defined	_	_	_	
2 by request	_	_	_	
6. queueing	.	.	.	- - - - - - - - - - - - - -
1 automatic	_	_	_	
2 by request	_	_	_	
7. record keeping	.	.	.	- - - - - - - - - - - - - -
1. log	.	.	.	
1 automatic	_	_	_	
2 by request	_	_	_	
2. journal	.	.	.	
1 automatic	_	_	_	
2 by request	_	_	_	
5.3. Design Change				
1. type	.	.	.	- - - - - - - - - - - - - -
1 transmission	_	_	_	
2 reception	_	_	_	
2. implementation	.	.	.	- - - - - - - - - - - - - -
1. on-line	.	.	.	
1 by transaction	_	_	_	
2 by software change	_	_	_	
2 off-line	_	_	_	

MMI Capability	E	U	N	Comment
6.0 DATA PROTECTION/SECURITY				
6.1. User Identification				
1 user code	___	___	___	_______________
2 station code	___	___	___	_______________
3 job code	___	___	___	_______________
4 project code	___	___	___	_______________
5. password	.	.	.	_______________
1. fixed	.	.	.	
1 assigned	___	___	___	
2 user-chosen	___	___	___	
2. changing	.	.	.	
1 assigned	___	___	___	
2 user-chosen	___	___	___	
6.2. Data Access				
1. user code	.	.	.	_______________
1 for file	___	___	___	
2 for data category	___	___	___	
3 other	___	___	___	
2. password	.	.	.	_______________
1 for file	___	___	___	
2 for data category	___	___	___	
3 other	___	___	___	
3. access record	.	.	.	_______________
1 for user	___	___	___	
2 for file	___	___	___	
3 for data category	___	___	___	
4 other	___	___	___	
6.3. Data Change				
1. user code	.	.	.	_______________
1 for file	___	___	___	
2 for data category	___	___	___	
3 other	___	___	___	
2. password	.	.	.	_______________
1 for file	___	___	___	
2 for data category	___	___	___	
3 other	___	___	___	
3. change record	.	.	.	(audit trail) ________
1 for user	___	___	___	
2 for file	___	___	___	
3 for data category	___	___	___	
4 other	___	___	___	
4. error prevention	.	.	.	_______________
1 data validation	___	___	___	
2 redundant entry	___	___	___	

MMI Capability	E	U	N	Comment
6.4. Loss Prevention				
1. reversible procedures	.	.	.	
1 CONFIRM				
2 BACKUP				
2. file protection	.	.	.	
1. SAVE	.	.	.	
1 automatic				
2 by request				
2. archive	.	.	.	
1 automatic				
2 by request				
3. data transmission	.	.	.	
1 parity check				
2 other				
6.5. Design Change				
1. type	.	.	.	
1 user identification				
2 data access				
3 data change				
4 data loss				
5 other				
2. implementation	.	.	.	
1. on-line	.	.	.	
1 by transaction				
2 by software change				
2 off-line				
3. change control	.	.	.	
1 data entry				
2 data display				
3 sequence control				
4 user guidance				
5 data transfer				
6 data protection				

APPENDIX B: DESIGN GUIDELINES FOR DATA ENTRY

DATA ENTRY/INPUT

Objectives:
Minimized input actions by user.
Low memory load on user.
Consistency of data entry transactions.
Compatibility of data entry with data display.
Flexibility for user control of data entry.

1.0* General

-1 When data entry is a significant task function, it should be accomplished via the user's primary display.

Example: Entry via typewriter is acceptable only if the typewriter itself, under computer control, is the primary display medium.

-2 Data entry transactions, and associated displays, should be designed so that the user can stay with one mode of entry as long as possible before having to shift to another.

Example: Shifts from lightpen to keyboard input and then back again should be minimized.

-3 Keyed inputs should always appear in the display.

Exception: Passwords and other secure entries.

Reference: Draft MIL-STD-1472C; item 5.15.3.8.4.2.

-4 Keyed data entry and data changes on an electronic display should generally be accomplished by direct character replacement, in which keyed inputs replace underscores or previous entries (including default values) in defined data fields.

Exception: Text entry.

* Decimal numbers refer to coding scheme of functional capabilities adopted for the MMI requirements checklist.

1.0 General (cont.)

-5 Whenever possible, data entry should be self-paced, depending upon the user's needs, attention span and time available, rather than computer processing or external events.

Comment: When self-pacing does not seem feasible, the general approach to task allocation and MMI design should be reconsidered.

-6 Data entry should not be slowed or paced by delays in control response; for normal operation control delays or lockouts should not exceed 20 milliseconds.

Reference: Draft MIL-STD-1472C; item 5.15.3.3.

-7 Data input should always require an explicit ENTER action, and not be accomplished as a side effect of some other action.

Example: Returning to a menu of control options should not by itself result in entry of data just keyed onto a display.

-8 An ENTER key should be explicitly labeled to indicate its function to the user.

Example: The ENTER key should not be labeled in terms of mechanism, such as CR or RETURN or XMIT.

-9 When a stored data item is changed (or deleted) by direct command entry without first being displayed, then both the old and new values should be displayed for user confirmation before the transaction is completed.

Comment: This practice will tend to prevent inadvertent change, and is particularly useful in protecting delete actions.

-10 Ideally, the length of individual data entries should not exceed 5-7 characters.

Exception: Textual material.

Comment: Longer items exceed the user's memory span, inducing errors in both data entry and data review.

1.0 General (cont.)

-11 When longer items must be entered, the item should be partitioned into shorter symbol groups for both entry and display.

Example: A 10-digit telephone number can be entered as three groups, ___-___-____.

-12 When portions of a long item are highly familiar or rendundant, ideally those should be entered last.

Exception: But not if this sequence would violate a functional requirement, such as initial keying of area code in telephone numbers.

Comment: This practice will reduce the load on the user's short-term memory.

-13 Minimize keying in data entry by abbreviation of lengthy inputs when that can be done without ambiguity.

Comment: Provide data validation routines and user interrogation as necessary to resolve any ambiguity that may arise.

-14 When abbreviated codes are used to shorten data entry, code values should be designed to be as distinctive as possible in order to avoid potentially confusing similarity.

Example: BOS vs. LAX is good; LAS vs. LAX is bad.

-15 When alphabetic data entry is required, restricted alphabetic sets should not be used.

Comment: Software might be provided to interrogate the user to resolve any input ambiguities resulting from hardware limitations; see Smith and Goodwin, 1971.

Reference: Smith, S. L. and Goodwin, N. C. Alphabetic data entry via the Touch-Tone pad: A comment. Human Factors, 1971, 13(2), 189-190.

1.0 General (cont.)

-16 Special characters used in data entry (, * = / etc.), particularly if used frequently, should be chosen insofar as possible so that the user will not have to shift from one case to another on the keyboard.

Comment: Conversely, keyboard designers should put frequently used special characters where they can be easily keyed.

-17 Entry of leading zeros should be optional for general numeric input.

Exception: Special cases such as entry of serial numbers or other numeric identifiers.

1.1 Position Designation (Cursor Control)

-1 Position designation on an electronic display should be accomplished by means of a movable cursor with distinctive visual features (shape, blink, etc.).

Exception: When position designation involves only selection among displayed alternatives, then some form of highlighting selected items might be used instead of a separately displayed cursor.

-2 If multiple cursors are used (e. g., one for alphanumeric entry, one for tracking, one for line drawing), they should be visually distinctive from one another.

-3 The cursor should be designed so that it does not obscure any other character displayed in the position designated by the cursor.

-4 When fine accuracy of positioning is required, as in some forms of graphic interaction, the displayed cursor should include a point designation feature.

1.1 Position Designation (cont.)

-5 Successful accomplishment of a position designation action should be signaled by immediate feedback to the user.

Example: Almost any consistently programmed display change will suffice, perhaps brightening or flashing a selected symbol; in some applications it may be desirable to provide an explicit message indicating that a selection has been made.

-6 Actual entry ("activation") of a designated position should be accomplished by an explicit user action distinct from cursor placement.

-7 If there is a predefined HOME position for the cursor, which is usually the case, that position should be consistent from one display to another.

Example: If HOME is in the upper left corner of an alphanumeric display, then the user may be disconcerted to find HOME in the center of a graphic display at the same work station.

Comment: The HOME position of the cursor should also be consistent in the different windows/sections of a partitioned display.

-8 For arbitrary position designation, the cursor control should permit both fast movement and accurate placement: rough positioning should take no more than 0.5 seconds for a displacement of 20-30 cm on the display; fine positioning may require incremental stepping of the cursor, or a control device incorporating a large control/display ratio for small displacements, or a selectable vernier mode of control use.

-9 The displayed cursor should be stable, i.e., should remain when it is placed until moved by the user (or computer) to another position.

-10 When cursor positioning is incremental by discrete steps, the step size of cursor movement should be consistent in both right and left directions, and both up and down directions.

-11 When displayed character size is variable, incremental cursor positioning should have a step size corresponding to the currently selected character size.

1.1 Position Designation (cont.)

-12 If proportional spacing is used for displayed text, the computer should be programmed to make necessary adjustments automatically when the cursor is being positioned for data entry or data change.

Comment: The user cannot be relied upon to do this accurately.

-13 Continuous position designation, such as used for input of track data, should be accomplished by continuously operable controls (e.g., thumb wheel for one dimension, joystick for two dimensions) rather than by incremental, discrete key actions.

-14 When position designation is the sole or prime means of data entry, as in selection of displayed alternatives, cursor placement should be accomplished by a direct-pointing device (e.g., lightpen) rather than by incremental stepping or slewing controls (keys, joystick, etc.).

-15 In selection of displayed alternatives, the acceptable area for cursor placement should be made as large as possible, including at least the area of the displayed label plus a half-character distance around the label.

-16 When position designation is combined with keyed data entry, cursor movement should be controlled at the keyboard (by function keys, "cat", joystick, etc.) rather than by a separately manipulated device (lightpen, "mouse", etc.).

-17 If multiple cursors are controlled by different devices, their separate controls should be compatible in operation.

-18 If multiple cursors are controlled by the same device, then a clear signal must be provided to indicate to the user which cursor is currently under control.

-19 On initial appearance of a data entry display the cursor should be placed automatically at the first character position of the first input field.

-20 Display formats for data input should be designed so as to minimize user actions required for cursor movement from one entry field to the next.

1.1 Position Designation (cont.)

-21 Sequential cursor positioning in predefined areas, such as displayed data entry fields, should be accomplished by programmable tab keys.

-22 Areas of a display not needed for data entry (such as labels and blank spaces) should be made inaccessible to the user, under computer control, so that the cursor does not have to be stepped through blank areas nor are they sensitive to pointing actions.

Comment: Mechanical overlays on the display should not be used for this purpose.

-23 User action confirming entry of multiple data items should result in input of all items, regardless of where the cursor is placed on the display.

1.2 Direction Designation

-1 When direction designation is based on already quantified data, then keyed entry should be used.

-2 When direction designation is based on graphic representation, then some "analog" means of entry should be provided, such as vector rotation on the display, and/or a suitably designed rotary switch.

Example: Heading estimation for displayed radar trails.

Exception: When only approximate direction designation is required, for just eight cardinal points, keyed entry can be used.

Comment: In matching graphic representation an analog entry device will prove both faster and more accurate; see Smith, 1962.

Reference: Smith, S. L. Angular estimation. Journal of Applied Psychology, 1962, 46, 240-246.

1.3 Text

(no guidelines presently available)

1.4 Data Forms

-1 Using a form-filling dialogue, entry of logically grouped items should be accomplished by a single, explicit action at the end, rather than requiring separate entry of each item.

-2 When multiple items are entered as a single transaction, as in form filling, the user should be allowed to RESTART, CANCEL, or BACKUP and change any item before taking a final ENTER action.

Reference: Draft MIL-STD-1472c; items 5.15.1.2.4, 5.15.3.8.4.1.

-3 Whenever possible, multiple data items should be entered without the need for special separators or delimiters, either by keying into predefined entry fields or by including simple spaces between sequentially keyed items.

-4 When a field delimiter must be used for data entry, a standard character should be adopted for that purpose; slash (/) is preferred.

-5 For all dialogue types involving prompting, data entries should be prompted explicitly by means of displayed labels for data entry fields and/or associated user guidance messages.

-6 Field labels should consistently indicate what data items are to be entered.

Example: A field labeled NAME should always require name entry, and not sometimes require something different like elevation.

-7 Implicit cues should be provided in form-filling dialogues to supplement explicit labels; special characters (e.g., underscores) should used to delineate each entry field.

-8 Field delineation cues should distinguish required (dashed underscore) from optional (dotted underscore) entries, and should indicate the maximum acceptable length of the entry.

Comment: Similar implicit cues can be provided when data entry is prompted by auditory displays; see Smith and Goodwin, 1970.

Reference: Smith, S. L. and Goodwin, N. C. Computer-generated speech and man-computer interaction. Human Factors, 1970, 12(2), 215-223.

-9 When item length is variable, the user should not have to justify an entry either right or left, and should not have to remove any unused underscores.

-10 When multiple items (especially those of variable length) will be entered by a skilled touch typist, each entry field should end with an extra (blank) character space; an auditory signal should be provided to alert the user if an entry is keyed into a blank space.

Comment: This will permit consistent use of tab keying to move from one field to the next.

-11 When entry fields are distributed across a display, a consistent format should be adopted for relating labels to delineated entry areas.

Example: The label might always be to the left of the field; or the label might always be immediately above and left-justified with the beginning of the field.

Comment: Such consistent practice will help the user distinguish labels from data in distributed displays.

-12 Labels for data entry fields should be distinctively worded, so that they will not be readily confused with data entries, labeled control options, guidance messages, or other displayed material.

1.4 Data Forms (cont.)

-13 In labeling data entry fields, only approved terms, codes and/or abbreviations should be used.

Comment: Do not create new jargon; if in doubt, pretest all proposed wording with a sample of qualified users.

-14 Labels for entry fields may incorporate additional cueing of data formats when that seems helpful.

Example: DATE(MDY)= __:__:__.

-15 When a dimensional unit ($, mph, km, etc.) is consistently associated with a particular data field, it should be displayed as part of the fixed label rather than entered by the user; when alternative dimensional descriptors are acceptable, then space should be provided in the data field for user entry of a unit designator.

-16 When data entry displays are crowded, auxiliary coding should be adopted to distinguish labels from data.

Example: A recommended standard is to display fixed, familiar labels in dim characters, with data entries bright.

-17 Display formats for data entry should be identical or compatible with formats used for display output, scanning and review of the same data items.

Comment: When a display format optimized for data entry seems unsuited for data display, or vice versa, some compromise format should be designed taking into account the relative functional importance of data entry and review in the user's task.

-18 When data entry involves transcription from source documents, in a form-filling dialogue displayed forms should match (or be compatible with) paper forms; in a question-and-answer dialogue, the sequence of entry should match the data sequence in source documents.

Comment: When paper forms are not optimal for data entry, consider revising the layout of the paper form; when data entries must follow an arbitrary sequence of external information (e.g., keying telephoned reservation data), some form of command language dialogue should be used instead of form filling, to identify each item as it is entered so that the user does not have to remember and re-order items.

1.4 Data Forms (cont.)

-19 If no source document or external information is involved, the ordering of multiple-item data entries should follow the logical sequence in which the user can be expected to think of them.

Comment: Alternatively, data entry can sometimes be made more efficient by placing all required fields before any optional fields.

1.5 Tabular Data

-1 When tabular formats are used for data entry, column labels should be left-justified with the leftmost position beginning column entries, especially when columns vary in width.

-2 Right-or left-justification of tabular data entries should be handled automatically by the computer; in particular, the user should not have to enter any leading blanks or other extraneous formatting characters.

-3 Numeric entries (e.g., dollars and cents), if entered as left-justified, should be automatically justified with respect to a fixed decimal point when a display of these data is regenerated for review by the user.

-4 For input of tabular data, when vertical repetition of entries is frequent the user should be provided a DITTO key to speed entry of duplicative data.

1.6 Graphic Data

(no guidelines presently available)

1.7 Data Validation

-1 Automatic data validation software should be incorporated to check any entry whose input and/or correct format or content is required for subsequent data processing.

Comment: Do not rely on the user always to make correct inputs.

-2 When required data entries have not been input, but can be deferred, data validation software should signal that omission to the user, permitting either immediate or delayed input of missing items.

-3 When entry of a required data item is deferred, the user should have to enter a special symbol in the data field to indicate that the item has been temporarily omitted rather than ignored.

-4 In a repetitive data entry task, data validation for one transaction should be completed, and the user allowed to correct errors, before another transaction can begin.

Comment: This is particularly important when the user is transcribing data from source documents, so that detected input errors can be corrected while the relevant document is still at hand.

-5 If item by item data validation within a multiple-entry transaction is provided, it should only be as a selectable option.

Comment: This capability may be helpful to a novice user, but it will tend to interrupt and slow a skilled user.

-6 When useful default values for data entry cannot be predicted by system designers, which is often the case, the user (or perhaps some authorized supervisor) should be provided a special transaction to define, change or remove default values for each data entry field.

-7 Currently defined default values should be displayed automatically in their appropriate data fields with initiation of a data entry transaction.

Comment: The user should not be expected to remember them.

1.7 Data Validation (cont.)

-8 User acceptance of a default value should be accomplished by simple means, such as by a single confirming key input or by tabbing past the default field.

Comment: Similar techniques should be used in tasks involving user review of stored data.

-9 In any data input transaction the user should be able to replace a default value with a different entry, without changing the current default definition.

1.8 Other Data Processing

-1 The user should not be required to enter "bookkeeping" data the computer could be programmed to know automatically.

Example: A user generally should not have to identify his work station to initiate a transaction, nor include other routine data such as transaction sequence codes.

Comment: Complicated data entry routines imposed in the supposed interest of security may hinder the user in achieving effective task performance; other means of ensuring data security should be considered.

-2 The user should not be required to enter redundant data already known to the computer.

Example: The user should not have to enter both an item name and identification code when either one defines the other.

Exception: As needed for resolving ambiguous entries, user training, or security, i. e., user identification.

Comment: Verification of stored data is usually better handled by review and confirmation rather than by re-entry.

-3 Data entries made in one transaction should be remembered by the system when relevant to another transaction, and displayed for review if appropriate.

Comment: The user should not have to enter such data again.

1.8 Other Data Processing (cont.)

-4 Whenever needed, automatic computation of derived data should be provided, so that the user does not have to calculate and enter any number that can be derived from data already entered in the system.

-5 Whenever needed, automatic cross-file updating should be provided, so that the user does not have to enter the same data twice.

Example: Assignment of aircraft to a mission should automatically indicate that commitment in squadron status files as well as in a mission assignment file.

1.9 Design Change

(no guidelines presently available)

APPENDIX C: DESIGN GUIDELINES FOR SEQUENCE CONTROL

SEQUENCE CONTROL

Objectives:
- Minimized control actions by user.
- Low memory load on user.
- Consistency of control actions.
- Compatibility of sequence control with user needs.
- Flexibility of sequence control.

3.0 General

-1 Flexible means of sequence control should be provided so that the user can accomplish necessary transactions involving data entry, processing, retrieval and transmission, or can obtain guidance as needed in connection with any transaction.

Example: In scanning a multi-page display the user should be able to go forward or back at will; if the MMI design permits only forward steps, so that the user must cycle through the entire display series to reach a previous page, that design is deficient.

Comment: Necessary transactions should be defined in task analysis prior to software design.

Reference: PR 4.0.*

-2 Control inputs should be simplified to the maximum extent possible, particularly for tasks involving real-time response, and should permit completion of a transaction sequence with the minimum number of control actions consistent with user abilities.

Example: The user should be able to print a display directly without having to take a series of other actions first, such as calling for the display to be filed, specifying a file name, then calling for a print of that named file.

Comment: The software designer should program the computer to handle intervening steps automatically, informing the user what has been done if that seems necessary.

Reference: MS 5.15.2.7.

---

* See note on references at the end of this appendix.

-3 The means of sequence control should be compatible with desired ends; frequent or urgent control actions should be easy to take, whereas potentially destructive control actions should be difficult so as to require explicit user attention.

-4 Sequence control should be compatible with user skills, permitting simple step-by-step control actions for beginners, and efficiently coded command entry by experienced users.

Comment: This will generally require a mix of dialogue types (see section 3.1).

-5 In most on-line information handling systems, sequence control should result from explicit user inputs rather than occur as an automatic consequence of computer processing.

Example: The computer should not interrupt user data entry to require immediate correction of any input error, but instead should wait until the user signals completion of the transaction.

Exception: Routine, repetitive transaction sequences in which successful completion of one may lead automatically to initiation of the next.

Exception: Automated process control applications where emergency conditions may take precedence over current user transactions.

Comment: In general, computer detection of problems with current user inputs can be negotiated at the conclusion of a transaction, before it is implemented; computer detection of other problems can be signaled by alarms or advisory messages, so that the user can choose when to deal with them.

See also: 1.0-7; 1.1-6; 1.4-1.

-6 Whenever possible, control inputs should be self-paced, depending upon the user's needs, attention span and time available, rather than computer processing or external events.

Comment: When self-pacing does not seem feasible, the general approach to task allocation and MMI design should be reconsidered.

See also: 1.0-5.

3.0 <u>General</u> (cont.)

-7 User control inputs should not be delayed or paced by delays in computer response; control delays or lockouts should not exceed 20 milliseconds.

Reference: MS 5.15.3.3.

See also: 1.0-6.

-8 Although the man-machine dialogue is necessarily limited by the computer, software design should insofar as possible permit initiative and control by the user; the software designer should anticipate all possible user actions and their consequences, and should provide appropriate options in every case.

Comment: In particular, a dialogue should never reach a dead end with no further action available to the user; if the user makes an input inappropriate (or unrecognizable) to current processing logic, the result should simply be an advisory message indicating the nature of the problem and the available options as to what can be done next.

Reference: PR 2.2.

-9 All control inputs made by the user should be acknowledged by the computer, either by their immediate execution, or else by some immediate message indicating that execution is in progress or deferred or that the control input requires correction or confirmation.

Example: In particular, the absence of computer response is <u>not</u> an acceptable means of indicating that a command is being processed.

Comment: "Immediate" as used here is subject to interpretation in relation to the response time requirements of different dialogue types.

Reference: EG 4.2.5; MS 5.15.3.4.

See also: Section 3.1.

-10 If further user inputs must be delayed pending completion of computer processing, the keyboard should be automatically locked until the user can begin a new transaction; keyboard lock should be accompanied by disappearance of the cursor from the display and (especially if infrequent) by some more specific indicator such as an auditory signal.

3.0 General (cont.)

-11 When execution of a control input is delayed, the computer should give the user some positive indication when processing is subsequently completed, the outcome, and the implied need for further user actions if any.

Reference: MS 5.15.1.3.4.c.

-12 Computer responses to control inputs should generally consist of changes in state or value of displayed elements affected by the control action, in an expected or natural form.

Reference: MS 5.15.3.4.

-13 In data entry tasks where input is usually accomplished as a single, occasional transaction, successful entry should be signaled by a confirmation message without removing any visual display of the entered data.

Comment: This follows a general principle of sequence control that the user should leave one transaction and choose the next by explicit command.

Reference: MS 5.15.1.2.6.d.

-14 In data entry tasks where input is usually repetitive, in a continuing sequence of transactions, successful entry should be signaled by regeneration of the data entry display, automatically removing the just entered data in preparation for the next entry.

Comment: This represents an exception to the general principle of sequence control by explicit user choice, in the interest of efficiency.

Reference: EG 4.2.10.

-15 Sequence control actions should be consistent in form and consequences throughout MMI design; similar means should be employed to accomplish similar ends, from one transaction to the next, and from one task to another.

Comment: In particular, there should be some standard, consistent routine for the user to initiate and complete task sequences.

3.0 General (cont.)

-16 If the consequences of a given control action will differ depending upon context established by a prior action, then some appropriate means of context definition should be displayed in advance to the user.

Comment: Do not rely on the user always to remember prior actions, nor to understand their current implications.

-17 The design of linked transaction sequences should be based on task analysis, i. e., should represent a logical unit or subtask from the viewpoint of the user.

Comment: A logical unit to the user is not necessarily the same as a logical unit of the computer software that mediates the transaction sequence.

Reference: PR 5.1.

-18 Displays should be designed so that portions relevant to sequence control are distinctive in position and/or format; relevant features include displayed options and any command entry area used to indicate control actions, plus advisory messages and other displayed items (titles, time signals, etc.) whose changes signal the results of control actions.

-19 When two or more users must interact with the system simultaneously, control inputs by one should not interfere with those of another.

Reference: MS 5.15.2.5.

-20 In instructional material and in on-line messages to the user, consistent terminology should be adopted to refer to control inputs.

Example: Various words and phrases might be used, such as "control input", "command entry", "instruction", "request", "function call", etc.; the standard adopted in these guidelines is to refer to general sequence control actions as "control inputs" and control inputs keyed onto the display as "command entries".

3.1 Dialogue Type

-1 Choice of dialogue type(s) and design of sequence control dialogue should take into account user characteristics and task requirements.

Example: When data entries must be made in arbitrary order, perhaps mixed with queries (as in making flight reservations), then some mixture of function keys and coded command entries will be required for effective operation, involving a moderately high level of user training.

Comment: The simple dictum is "Know the user"; if user characteristics are variable, which is often the case, then a variety of dialogue types should be provided.

-2 The speed of computer response to user inputs should be appropriate to the type of dialogue; in general, the response to menu selections, function keys, and most inputs during graphic interaction should be immediate.

Example: Maximum acceptable computer response for menu selection by lightpen is 1.0 second; for key activation is 0.1 second; for cursor positioning by lightpen (as in graphic line drawing) 0.1 second.

Comment: If computer response time will be slow, other dialogue types should be considered by the software designer.

Reference: Miller, R. B. Response time in man-computer conversational transactions. In Proceedings of the AFIPS Fall Joint Computer Conference, 1968, 267-277.

-3 The speed of computer response to user inputs should be appropriate to the transaction involved; in general the response should be faster for those transactions perceived by the user to be simple.

Example: Computer response to a predictable command, such as NEXT PAGE, should be within 0.5-1.0 second; response to other simple commands should be within 2.0 second; error messages should be displayed within 2-4 second.

Reference: Miller, 1968.

3.1.1 Question and Answer

-4 Question-and-answer dialogue should be considered primarily for routine data entry tasks, where data items are known and their ordering can be constrained, where the user will have little or no training, and where computer response is expected to be moderately fast.

Comment: Brief question-and-answer sequences can be used to supplement other dialogue types for special purposes, such as for log-on routines, or for resolving ambiguous control inputs or data entries.

3.1.2 Form Filling

-5 Form-filling dialogue should be considered when some flexibility in data entry is needed, such as the inclusion of optional as well as required items, where users will have moderate training, and/or where computer response may be slow.

See also: Section 1.4.

3.1.3 Menu Selection

-6 Menu selection should be considered for tasks such as scheduling and monitoring that involve little entry of arbitrary data, where users may have relatively little training, and where computer response is expected to be fast.

Comment: Menu selection is, of course, a generally good means of mediating control inputs by untrained users in conjunction with other dialogue types for other task requirements.

-7 When menu selection is the primary means of sequence control, and especially if extensive lists of control options must be displayed, then selection should be accomplished by direct pointing (e. g., by lightpen).

-8 If menu selection is handled by pointing, a dual mode of activation should be provided, the first action to designate (position a cursor on) the selected option, followed by a separate action to make an explicit control input.

See also: 3.0-5; 1.0-7.

3.1.3 Menu Selection (cont.)

-9 When menu selection is a secondary (occasional) means of control input, and/or only short option lists are needed, then selection may be accomplished by keyed entry of corresponding codes, or by other means such as programmed multi-function keys labeled in the display margin.

-10 When menu selection is accomplished by code, that code should be keyed into a standard command entry area (window) in a fixed location on all displays.

Comment: For experienced users, coded menu selections can be keyed in a standard area identified only by its consistent location and use; if the system is designed primarily for novice users, that entry area should be given an appropriate label such as ENTER CHOICE HERE: ___.

Reference: PR 4.6.3; MS 5.15.4.7.1.d.

-11 Menu options should be worded so as to permit direct selection of any option as an acceptable control input, either by pointing or by code entry; options should <u>not</u> be worded so as to imply a question requiring a YES/NO answer.

Example: +PRINT is acceptable; PRINT? is not.

Reference: PR 4.6.8.

-12 When control inputs will be selected from a discrete set of options, then those options should be displayed at the time of selection.

Reference: MS 5.15.3.5.

-13 If menu selection is used in conjunction with (as an alternative to) command language, then displayed control options should be worded in terms of recognized commands or command elements; where appropriate, sequences of menu selections should be displayed in an accumulator until the user signals entry of a completely composed command.

Comment: This practice will speed the transition for a novice user, relying initially on sequential menu selection, to become an experienced user composing coherent commands without such aid.

3.1.3 Menu Selection (cont.)

-14 If menu selections must be made by keyed codes, options should be coded by the initial letter (or first several letters) of their displayed labels rather than by more arbitrary numeric codes.

Exception: Option selection from long lists, where line number might be an acceptable code alternative to keying an entire item.

Comment: Letters are easier than numbers for touch typists; options can be re-ordered on a menu without changing letter codes; it is easier to memorize meaningful names than numbers, and so letter codes can facilitate a potential transition from menu selection to command language when those two dialogue types are used together.

Reference: Palme, J. A human-computer interface for non-computer specialists. Software -- Practice and Experience, 1979, 9, 741-747.

-15 If letter codes are used to make menu selections, then insofar as possible those codes should be used consistently in designating options at different steps in a transaction sequence.

Example: The same action should not be given different names and hence different codes (F=FORWARD and N=NEXT); the same code should not be given to different actions (Q=QUIT and Q=QUEUE).

-16 If menu options are included as a portion of a display intended also for data review and/or data entry, which is often a practical design approach, the displayed labels for control input should incorporate some consistent distinguishing feature to indicate their special function.

Example: All control options might be displayed beginning with a special symbol such as a plus sign.

-17 Displayed menu options should be listed in a logical order; if no logical structure is apparent, then options should be displayed in order of their expected frequency of use, with the most frequent listed first.

Comment: If the first menu option is always the most likely choice, then for some applications it may be useful to provide an automatic default to the first item for efficiency of sequence control.

Reference: PR 4.6.6; Palme, 1979.

-18 Displayed menu lists should be formatted to indicate the hierarchic structure of logically related groups of options, rather than as an undifferentiated string of alternatives.

-19 If menu options are grouped in logical subunits, those groups should be displayed in order of their expected frequency of use.

Reference: PR 4.6.6.

-20 If menu options are grouped in logical subunits, each group should be given a descriptive label which is distinctive in format from the labels of the control options themselves.

Comment: Although this practice might be considered wasteful of display space, it will help provide user guidance; moreover, careful selection of group labels may serve to reduce the number of words needed for individual option labels.

Reference: MS 5.15.2.10.

-21 A displayed menu should include only options appropriate at that particular step in a transaction sequence, and for the particular user.

Example: Displayed file directories should contain only those files actually available to the user.

Example: An UPDATE option should be offered only if the user has update rights for the particular data file being used.

Exception: Menu displays for a system still under development might indicate future options not yet implemented, but those options should be specially designated in some way.

3.1.3 Menu Selection (cont.)

-22 Insofar as possible a displayed menu should include all options appropriate at that particular step in a transaction sequence.

Exception: A familiar set of general control options always available may be omitted from individual displays, and accessed as needed by a +OPTIONS input.

See also: Section 3.2.

-23 When option selections must be made from a long list, and not all options can be displayed at once, a hierarchic sequence of menu selections should be provided rather than one long multi-page menu.

Exception: Where a long list is already structured for other purposes, such as a list of customers, a parts inventory, a file directory, etc., it might be reasonable to require the user to scan multiple display pages to find a particular item; even in such cases, however, an imposed structure for sequential access may prove more efficient.

Comment: Multi-page option lists will generally hinder learning and use; the software designer can usually devise some means of logical segmentation to permit several sequential selections among few alternatives instead of a single difficult selection among many.

-24 When the user must step through a sequence of menus to make a selection, the hierarchic structure should be designed, insofar as possible within the constraints of display space, to minimize the number of steps required.

Comment: This represents a trade-off against the previous guideline; where space permits, it may be desirable to display further (lower) choices in the hierarchic structure, to give the user a deeper view of the structure and permit direct selection of specific lower-level options.

Reference: MS 5.15.2.2.

-25 When hierarchic menus are provided, they should be designed to permit the user immediate access to critical or frequently selected options.

Reference: MS 5.15.2.2.

3.1.3 Menu Selection (cont.)

-26 When hierarchic menus are provided, the user should be given some displayed indication of current position in the menu structure.

Reference: MS 5.15.2.2.

-27 When hierarchic menus are provided, care should be taken to ensure compatible display formats at each level.

Reference: MS 5.15.2.2.

-28 Menus provided in different displays should be designed so that option lists are compatible in terminology and ordering.

Example: If +PRINT is the last option in one menu, the same print option should not be worded +COPY at the beginning of another menu.

-29 When a displayed control option has been selected and entered, if there is no immediately observable natural response the selected option label should be highlighted in some way (e. g., brightening or inverse video) to indicate computer acknowledgment.

Reference: MS 5.15.1.4.a.

See also: 1.1-5.

3.1.4 Function Keys

-30 Function keys should be considered for tasks requiring only a limited number of control inputs, or in conjunction with other dialogue types as a ready means of accomplishing critical inputs which must be made quickly without syntax error.

Reference: MS 5.15.3.7.

3.1.4 Function Keys (cont.)

-31 Function keys should be considered as a means of accomplishing frequently required control inputs.

Example: ENTER, PRINT, NEXT PAGE, PREV PAGE, OPTIONS, etc.

Comment: When generally used options are always implicitly available via function keys, they need not be included in displayed menus.

See also: 3.1-22; Section 3.2).

-32 Function keys should be used as a means of permitting interim control inputs, i. e., for control actions taken before the completion of a transaction.

Example: TAB, DITTO, DEFAULT, HELP, etc.

-33 Function keys should be labeled informatively to designate the function they perform; labels should be sufficiently different from one another to prevent user confusion.

Example: Log-on should not be initiated by a key labeled PANIC.

Reference: MS 5.15.2.10.

-34 If a function is continuously available, a single label should be on the key.

-35 If a key is used for different functions depending upon defined operational mode, then alternate self-illuminated labels should be provided on the key to indicate which function is current.

Comment: In these circumstances, it is preferable that only the currently available function is visible, so that the labels on a group of keys will show what can be done at any point rather than what has been done.

-36 When a function key performs different functions in different operational modes, those functions should be made as consistent as possible.

Example: A key labeled RESET should not be used to dump data in one mode, save data in another, and signal task completion in a third.

3.1.4 Function Keys (cont.)

-37 If the function of a key is specific to a particular step in the transaction sequence, then the current function should be indicated by an appropriate guidance message on the user's display.

-38 Function keys (and other devices) not needed for current inputs should be temporarily disabled under computer control at any step in a transaction sequence; mechanical overlays manipulated by the user should not be used for this purpose.

Comment: If the user selects a function key that is invalid at a particular step in a transaction sequence, no action should result except display of an advisory message indicating what functions are appropriate at that point.

Reference: PR 4.12.4.5; MS 5.15.3.8.4.3.

-39 Function keys should be grouped in distinctive locations on the keyboard to facilitate their learning and use; frequently used function keys should be placed in the most convenient locations.

Comment: It is preferable that frequently used keys not require double (control/shift) keying.

Reference: MS 5.15 4.7.1.d.

-40 The layout of function keys should be compatible with their importance; keys for emergency functions should have a prominent position and distinctive coding (e. g., size and/or color); keys with potentially disruptive consequences should be physically protected.

-41 Function keys should require only single activation to accomplish their function, and should not change function with repeated activation.

Example: Log-on should not be initiated by pressing a PANIC key twice.

-42 When function key activation does not result in any immediately observable natural response, the user should be given some signal to indicate computer acknowledgment.

Comment: Temporary illumination of the function key would suffice, if key illumination is not used to signal available options; otherwise a displayed advisory message should be used.

3.1.5 Command Language

-43 Command language dialogue should be considered for tasks involving a wide range of user inputs, where users may be highly trained in the interests of achieving efficient performance, and where computer response is expected to be relatively fast.

Comment: Command language should also be considered for data entry in arbitrary sequence.

See also: 1.4-19).

-44 When command language is used for control input, an appropriate entry area should be provided in a consistent location on every display, preferably at the bottom if the cursor can be conveniently moved there.

Comment: Adjacent to the command entry area there should be another defined display window used for prompting control input, for recapitulation of command sequences (with scrolling to permit extended review), and to mediate question-and-answer dialogue sequences (i. e., prompts and responses to prompts).

Reference: MS 5.15.4.7.1.d.

-45 The words chosen for a command language should reflect the user's point of view and not the programmer's, corresponding consistently with the user's operational language, incorporating whatever jargon is common on the job.

Reference: EG 4.2.12; EG 4.2.13; MS 5.15.1.2.6.f.

-46 All words in a command language, and their abbreviations, should be consistently used and standardized in meaning from one transaction to another and from one task to another.

Example: Do not use EDIT in one place, MODIFY in another, UPDATE in a third, all referring to the same kind of action.

Reference: EG 4.2.9; EG 4.2.13; MS 5.15.2.6.

3.1.5 Command Language (cont.)

-47 Words in a command language should be chosen insofar as possible to be distinctive from one another in order to prevent user confusion.

Example: Do not label two commands DISPLAY and VIEW, when one permits editing displayed material and one does not.

Reference: MS 5.15.2.10.

-48 A command language should provide flexibility, permitting the user to assign personal names to files, frequently used command sequences, etc.

-49 A command language should be supported by whatever computer processing is necessary so that the user can manipulate data without concern for internal storage and retrieval mechanisms.

Example: The user should be able to request display of a file by name alone, without having to enter any further information such as file location.

-50 The user should be able to request prompts as necessary to determine required parameters in a command entry, or to determine available options for an appropriate next command entry.

-51 When command entries are prompted automatically, it should be possible for an experienced user to key a series of commands at one time ("command stacking") so as to shortcut the prompting sequence.

See also: Section 3.2.

-52 Insofar as possible, the user should not be required to provide punctuation in command entries.

-53 If a delimiter is required to distinguish optional parameters, or the separate keyed entries in a stacked command, a standard symbol should be used consistently for that purpose, preferably the same symbol (slash) used to separate a series of data entries.

See also: 1.4-4; 3.2-18.

3.1.5 Command Language (cont.)

-54 Neither the user nor the computer program should have to distinguish between single and multiple blanks in a command entry.

Comment: People cannot be relied upon to pay careful attention to such details; the computer should handle them automatically, e. g., ensuring that two spaces follow every period in text entry, adding spaces needed to justify lines of text, etc.

-55 When command entries are subject to misinterpretation (as in the case of voice input), or when an interpreted command may have disruptive consequences, the user should be given an opportunity to review and confirm a displayed interpretation of the command before it is executed.

3.1.6 Query Language

-56 Query language dialogue should be considered as a specialized sub-category of general command language for tasks emphasizing unpredictable information retrieval (as in many analysis and planning tasks), with moderately trained users and fast computer response.

3.1.7 Natural Language

-57 Natural language dialogue should not be considered for information system design at this time; natural language may find future use in applications where task requirements are broad ranging and poorly defined, where little user training can be provided, and where computer response will be fast.

Comment: Computer processing of natural language is now being developed on an experimental basis; for current applications where task requirements are well defined, other types of dialogue will prove more efficient.

3.1.8 Graphic Interaction

-58 Graphic interaction should be considered as a supplement to other forms of man-machine dialogue where special task requirements exist; effective implementation of a full range of graphic capabilities will require a high level of user training and very fast computer response.

3.2 Transaction Selection

-1 The sequence of transaction selections should generally be dictated by the user's choices and not by internal computer processing constraints.

Comment: In some cases this means that the computer may have to store current inputs until they become relevant to subsequent data processing.

Reference: PR 4.6.7.

See also: 3.0-1; 3.0-5; 3.0-8.

-2 An initial menu of control options should always be available for user selection, to serve as a "home base" or consistent starting point for control inputs at the beginning of a transaction sequence.

Comment: Such a starting point is helpful even when all dialogue is user-initiated; this capability can be implemented as an OPTIONS function key, or as an explicit control option on every display, or as a generally available implicit option, or as a consistent default for a null control input.

Reference: PR 3.3.16.

-3 The general OPTIONS display should show primary control inputs grouped, labeled and ordered in terms of their logical function, frequency and criticality of use, following the guidelines provided for menu selection.

See also: Section 3.1.3.

-4 The user should be able to make at least some sequence control inputs directly at any step in a transaction sequence (i. e., from any display frame) without having to return to a general options display.

Comment: In particular, the user should not have to remember data or control codes given on one display for later input on a different display.

-5 Information should be provided the user concerning control options specifically appropriate at any step in a transaction sequence, either incorporated in the display or else available through a request for HELP.

Reference: MS 5.15.3.5.

-6 Control options that are generally available at any step in a transaction sequence may be treated as implicit options, i. e., need not be included in a display of step-specific options; frequently used implicit options should be input by function keys, others by coded command entry.

See also: 3.1-31.

-7 When selection among displayed options is to be accomplished by pointing, the cursor should be placed automatically on the first (most likely) option at initial display generation.

-8 When selection among displayed options is to be accomplished by keyed entry of a corresponding code, the cursor should be placed automatically in the command entry area at initial display generation.

Reference: PR 4.7.1.

-9 When displayed options can be selected by code entry, the code associated with each option should be included on the display in some consistent, identifiable manner; in many applications an equal sign can be used for this purpose.

Example: N=NEXT PAGE, P=PREV PAGE, etc.

-10 The wording of step-specific control options should reflect the current concerns and likely questions of the user at that step in a transaction sequence.

Reference: MS 5.15.2.10.d.

-11 The user should not be offered control options that he cannot take.

See also: 3.1-21.

-12 If a default value for a null control input is defined for any step in a transaction sequence, that value should be indicated in the display of step-specific control options.

Reference: EG 4.2.4.

3.2 Transaction Selection (cont.)

-13 If control input is accomplished by command entry, then the user should have some consistent means to request prompting for options or control parameter values not already shown on the display.

Example: Keying a question mark in the command entry area would be a satisfactory method of requesting prompts, or else using an explicitly labeled HELP function key.

-14 At any step in a defined transaction sequence, if specific control options are not displayed then a standard command should be provided so that the user can continue to the next step.

Example: Either NEXT, STEP or CONTINUE might be suitable names for this function.

Exception: If data entry is involved, then an explicit ENTER command should be used.

Reference: PR 4.11.

-15 When control input involves command entry, or else code entries form a sequence of menus, then the user should be permitted to key a sequence of codes for option selection as a single "stacked" command.

Example: In particular, the user should be able to enter stacked commands from the initial menu of general OPTIONS, so that an experienced user can make any specific control input the first step in a transaction sequence.

Example: Command stacking may be helpful when a user is being prompted to enter a series of parameter values, and knows in advance what several succeeding prompts will request and what values to enter.

Comment: Command stacking will permit a transition from simple step-by-step control input by novice users, as in menu selection and question-and-answer dialogues, to the entry of extended command-language statements by experienced users; command stacking is especially helpful in time-shared systems where computer response to any user input may be slow.

Reference: EG 6.2; EG 6.2.1; PR 2.6; PR 4.7.3; Palme, 1979.

See also: 3.1-51.

3.2 Transaction Selection (cont.)

-16 In command stacking, user inputs should be in the same order as they would normally be made in a succession of separate command entry actions.

Reference: EG 6.2.1.

-17 In command stacking, acceptable user inputs should include command names or their abbreviations or defined codes; if stacked command inputs are potentially ambiguous, the computer should display the interpreted command sequence for user correction or confirmation.

Reference: EG 6.2.1.

-18 In command stacking, a standard symbol should be used to separate command entries, preferably the same symbol (slash) used as a delimiter for sequential data entries.

Reference: EG 6.2.1.

See also: 1.4-4; 3.1-53.

-19 If a stacked command results in only partial completion of a menu selection sequence, i. e., further user selections must be made, then the appropriate next menu display should be presented to guide completion of control input.

Reference: PR 4.7.3.

-20 Flexibility in transaction selection should be provided by permitting the user to assign a single name to a defined series of control inputs, and then use this new "macro" for subsequent command entry.

Comment: In this way the user can make frequently required but complicated tasks easier to accomplish, when the software designer has failed to anticipate the particular need.

3.3 Interrupt

-1 Flexibility in control should be provided by permitting the user to interrupt, defer or abort a current transaction sequence, in consistently defined ways appropriate to specific task requirements.

Comment: Provision of flexible interrupt capabilities for the user will generally require some sort of suspense file or other buffering in software design; such capabilities are valuable, however, both in permitting the user to change his mind and in permitting the computer to require user confirmation of potentially destructive entries.

Reference: PR 3.3.16; PR 3.3.17.

-2 Differently named options should be provided to accomplish different degrees of interruption in sequence control.

Comment: As a negative example, it would not be good design practice to provide a single ESCAPE key which has different effects depending upon whether it is pushed once or twice; the user may be confused by such expedients, and uncertain about what action has been taken and its consequences.

-3 If appropriate to sequence control, a CANCEL option should be provided, which will have the consistent effect of regenerating the current display without processing any interim changes made by the user.

Comment: In effect, interim entries would be erased.

-4 If appropriate to sequence control, a BACKUP option should be provided, which will have the consistent effect of returning to the display entered in the last previous transaction; BACKUP implies cancellation of any interim entries made in a pending transaction.

Comment: Such a BACKUP capability will generally prove feasible only in the software design of well-defined transaction sequences, but will prove helpful when it can be provided.

Reference: MS 5.15.1.2.5.

See also: 1.4-2.

3.3 Interrupt (cont.)

-5 If appropriate to sequence control, a RESTART option should be provided, which will have the consistent effect of returning to the first display in a defined transaction sequence, permitting the user to review a sequence of entries and make necessary changes; RESTART implies cancellation of any interim entries made in a pending transaction.

Comment: As an extension of the BACKUP capability, RESTART is useful only in well-defined transaction sequences such as step-by-step data entry in a question-and-answer dialogue.

-6 If appropriate to sequence control, an ABORT option should be provided, which will have the consistent effect of cancelling all entries in a defined transaction sequence; when data entries or changes will be nullified by an ABORT action, the user should be asked in an advisory message to CONFIRM the ABORT.

Comment: An ABORT action, combining the functions of RESTART and CANCEL, is again relevant only to well-defined transaction sequences, specifically those with a recognized beginning.

-7 If appropriate to sequence control, an END option should be provided, which will have the consistent effect of concluding a repetitive transaction sequence and returning control to a general OPTIONS menu.

Example: In routine data entry, where the end of one transaction is designed to lead automatically to the beginning of the next transaction, the user needs some control input such as END to signal when a batch of transactions has been completed.

Reference: EG 4.2.10.

See also: 3.0-14.

3.4 Context Definition

-1 Sequence control software should be designed to maintain context for the user throughout the series of transactions comprising a task, recapitulating previous inputs affecting present actions, and indicating currently available options where appropriate.

See also: 1.8-3; 3.0-16.

-2 In tasks where transaction sequences are variable, the user should be able to request a displayed list of prior entries if needed to determine present status.

Comment: Such a capability may not be needed for routine transactions if they are designed in such a way that each step identifies its predecessors explicitly, although even in those circumstances a user may be distracted and at least momentarily become confused.

Reference: EG 4.2.7.

-3 Insofar as possible, sequence control software should be designed to carry forward a representation of the user's knowledge base and current activities; the user should not have to re-enter previously entered data relevant to current control inputs.

Example: If data have just been stored in a named file, then the user should be able to request a printout of that file without having to re-enter its name.

Exception: If transactions involving contextual interpretation would have destructive effects (e. g., data deletion), then the interpreted command should be displayed first for user confirmation.

Comment: The software logic supporting contextual interpretation of control inputs need not be perfect in order to be helpful; when ambiguity results, it may still be easier for the user occasionally to review and correct an interpreted command than always to generate a complete command initially.

Reference: PR 2.3; MS 5.15.2.9.

3.4 Context Definition (cont.)

-4 When context for sequence control is established in terms of a defined operational mode, then some means should be provided to remind the user of the current mode and other pertinent information.

Example: If text is displayed in an editing mode, then a caption might indicate EDIT as well as the name of the displayed text; if an INSERT mode is selected for text editing, then some further displayed signal should be provided.

Reference: EG 4.2.1.

-5 The current value of any control parameter(s) currently operative should be displayed for user reference.

Comment: This practice is helpful even when the user selects all parameters himself, since he may well forget them, particularly if his task activities are interrupted.

Reference: MS 5.15.3.5.b.

-6 Whatever information is given the user to provide context for sequence control should be distinctive in location and format, and consistently displayed from one transaction to the next.

Reference: MS 5.15.3.6; MS 5.15.4.5.

3.5 Error Management

-1 The computer software should not induce errors in sequence control.

Example: If the user selects a function key that is invalid at a particular step in a transaction sequence, no action should result except display of an advisory message indicating what functions are appropriate at that point.

Reference: PR 4.12.4.5.

See also: 3.1-38.

-2 The user should be able to edit an extended command during its composition, by backspacing and rekeying, before taking an explicit action to ENTER the command.

Reference: EG 5.4.

3.5 Error Management (cont.)

-3 If an element of a command entry is not recognized, or logically inappropriate, sequence control software should prompt the user to correct that element without having to re-enter the entire command.

Example: A faulty command can be retained in the command entry area of the display, with the cursor automatically positioned at the incorrect item, plus an advisory message describing the problem.

Reference: EG 4.2.2; EG 4.2.3; MS 5.15.1.2.6.b.

-4 If an error is detected in a stacked series of command entries, it may help the user if the commands are executed to the point of error, or it may not; software design should be consistent in this regard.

Reference: EG 5.6; PR 4.7.3.

-5 If only a portion of a stacked command can be executed, that problem should be indicated to the user with appropriate guidance to permit completion of the control input.

See also: 3.2-19.

-6 The ENTER action for command entry should be the same as that for data entry; direct selection of menu options should also require some explicit ENTER action.

See also: 1.0-7; 1.1-6; 3.0-5.

-7 When a user completes correction of an error, whether of a command entry or data entry, the user should be required to take an explicit action to re-enter corrected inputs; the new ENTER action should be the same as whatever action was appropriate to make that input originally.

Reference: PR 4.12.4.6.

-8 When a default value is included in command entry, it may be helpful to recapitulate the command in its fully interpreted form for user confirmation; if this practice is followed, it should be done consistently.

-9 When a command entry is subject to misinterpretation, the user should be asked to review a displayed interpretation for correction or confirmation.

See also: 3.1-55.

3.5 Error Management (cont.)

-10 When a control input will cause any extensive change in stored data, procedures and/or system operation, and particularly if that change cannot be easily reversed, the user should be notified and required to confirm the action before it is implemented.

-11 The prompt for CONFIRM action should be worded in such a way that any potential data loss is clearly stated.

Example: CONFIRM DELETION OF ENTIRE AIRFIELD FILE may be adequate warning; CONFIRM DELETE ACTION is not.

-12 User confirmation of a control input or data entry should be accomplished with an explicitly labeled CONFIRM function key.

Comment: Confirmation should not be accomplished by pushing some other key twice.

See also: 3.1-33; 3.1-41.

-13 When the user signals that he is ready to log off, sequence control software should check pending transactions and, if data loss seems probable, should display an advisory message requesting confirmation.

Example: CURRENT DATA ENTRIES HAVE NOT BEEN FILED; SAVE IF NEEDED, BEFORE CONFIRMING LOGOFF.

Comment: The user will sometimes suppose that a job is done before he has taken necessary final actions.

-14 When a data entry transaction has been completed and errors detected, sequence control logic should permit direct, immediate correction by the user.

Comment: It is helpful to correct data entry errors at the source, i. e., while the entry is still in mind and source documents still at hand.

Reference: PR 2.5.

See also: 1.7-4.

3.5 Error Management (cont.)

-15 The user should be able to return easily to previous steps in a transaction sequence in order to correct an error or make any other desired change.

Reference: MS 5.15.1.2.5.

See also: Section 3.3.

-16 When considerations of data security do not prohibit, the user should be able to change any data that are currently displayed.

Comment: The user should not have to specify that he wants to make changes in advance of calling for a display; that requirement may be simpler for the software designer, but is confusing to the user.

3.6 Alarms

-1 In many applications, particularly those involving monitoring and process control, the user should be permitted to define conditions, in terms of variables and categories, that will result in automatic generation of alarm messages.

Example: The nurse in charge of an intensive care monitoring station might need to specify for each patient warning signals when blood pressure ("variable") exceeds or falls below defined levels ("categories").

-2 Alarm signals and messages may take a variety of forms, but should be distinctive and consistent for each class of events; the user may be permitted to define the nature of each alarm as well as its initiating event.

-3 The user should be provided a standard means of acknowledging and turning off non-critical alarm signals.

Example: A function key labeled ALARM ACK would suffice for that purpose.

Reference: MS 5.15.4.7.1.a.

-4 The user may be required to take more complicated actions in order to respond to critical alarms, and to acknowledge special alarms in special ways.

3.7 Design Change

(no guidelines presently available)

Note on References

A complete collection of MMI design guidelines might eventually fill a book. With hundreds of guidelines to consider, the referencing of source material and the cross-referencing of other guidelines could become cumbersome. In this appendix a first attempt has been made to deal with the referencing problem.

References to special source documents are given in full. References to general sources are given in abbreviated form, with a two-letter code followed by the paragraph or item number in the referenced document. Three general sources are referenced in this way:

EG = Engel, S. E. and Granda, R. E. Guidelines for Man/Display Interfaces, Technical Report TR 00.2720. Poughkeepsie, New York: IBM, December 1975.

PR = Pew, R. W. and Rollins, A. M. Dialog Specification Procedures, Report 3129 (revised). Cambridge, Massachusetts: Bolt Beranek and Newman, 1975.

MS = currently proposed draft revisions to MIL-STD-1472B. Military Standard: Human Engineering Design Criteria for Military Systems, Equipment and Facilities. Washington: Department of Defense, 31 December 1974.